Calibration of a Mobile Robot with Application to Visual Navigation
Zhengyou Zhang Olivier D. Faugeras
INRIA Sophia-Antipolis
2004 Route des Lucioles
06565 Valbonne FRANCE

Abstract
The problem of the calibration of a mobile robot between its stereo and odometric coordinate systems is addressed. A well-known equation is used to recover the rotation axis of the mobile robot in the stereo system, using information on motion from stereo. The calibration can be done with some simple maneuvers (pure rotation and pure translation). A motion estimation algorithm from stereo is used to determine the displacement in the stereo coordinate system. With the calibration result, we can transform the displacement that the robot should perform in the stereo reference to achieve its goal, into the robot’s internal commands. As a sample application, a complete visual navigation loop is demonstrated.

Keywords: Calibration, Visual navigation, Mobile robot, Robot vision, Motion from stereo.

1 Introduction
In Robotics, the calibration problem is very important in the application of research results. There is a variety of calibration problems for stereo [3,7] and for eye-hand systems [9]. A mobile robot must include sensing (vision, sonar) and locomotion [5], but we must know the relationship between the coordinate systems corresponding to each perceptual mode in order to combine the different sources. In this article, we address the calibration problem between the stereo system and the odometric system of the INRIA mobile robot. We call it the calibration problem for navigation systems or “eye-wheel” systems. It is essential for many applications.

This paper is organised as follows. Section 2 gives a brief presentation of the INRIA Mobile Robot. Section 3 addresses the Calibration problem. Section 4 describes how to estimate motion parameters based on visual information from stereo. Section 5 shows how to plan the trajectory for some simple navigation problems. As an example of the proposed calibration technique, we describe in Section 6 a complete visually guided navigation demonstration.

2 The INRIA Mobile Robot
The INRIA mobile robot (see Figure 1) is, for the moment, dedicated to navigating in an indoor environment and to making visual maps of this environment. The robot is an autonomous vehicle with two driving wheels and two passive rollers. Each of the two driving wheels is independently controlled by a motor, so that the robot can translate and rotate. The following is a fundamental internal command of the robot [6]:

\[
\text{MOVE } P \quad \text{RC} = D_r, D_l, P = v
\]

where \(D_r, D_l\) and \(v\) are all integers. \(D_r\) (resp. \(D_l\)) is the rotation of the right (resp. left) wheel (in millimeters); positive rotations correspond to forward motion of the wheel, negative to backward motion. \(v\) controls the execution time of the command by controlling the number of cycles taken to execute this command. In order to simplify the rotation command, the distance between the two driving wheels is adjusted to \((1800/\pi)\) mm. Here are some examples:

\[
\begin{align*}
\text{MOVE } P \quad \text{RC} &= -150, -150, P = 150 \quad \text{makes the robot roll forward 150 mm} \\
\text{MOVE } P \quad \text{RC} &= 200, 200, P = 200 \quad \text{makes the robot roll backward 200 mm} \\
\text{MOVE } P \quad \text{RC} &= -60, +60, P = 120 \quad \text{makes the robot rotate to the left 12 degrees} \\
\text{MOVE } P \quad \text{RC} &= 0, -120, P = 120 \quad \text{makes the robot rotate to the right 12 degrees with the right wheel fixed}
\end{align*}
\]

Three CCD cameras are mounted on the robot. The trinocular vision system builds a local 3D visual map of the environment. The details of this system can be found in [1]. The robot is also equipped with an ultrasonic sensor, but in this paper, the vision system is the only information source for navigation.

3 The Calibration Problem
The motion of a mobile robot is equivalent to the motion of the frame of reference of the stereo system (see Figure 2).
This reference frame is rigidly attached to the mobile robot and its numerical parameters are determined in the camera calibration phase [3].

The problem of calibration in an eye-wheel system is to determine the transformation between the stereo coordinate and odometric coordinate system. Solving this problem is important when we want to navigate a robot using visual information from stereo. Figure 3 shows the relation between the stereo and odometric coordinate systems. We know \((OXY)\) with respect to the world coordinate system by stereo calibration. In the following, we present a technique to calibrate the eye-wheel system by using a motion from stereo algorithm as described below. In other words, we should recover \(\vec{OC}\) and \(\vec{Cy}\) as shown in Figure 3, where \(\vec{OC}\) is the center of rotation and \(\vec{Cy}\) is the translation direction of the robot in the stereo coordinate system.

It is well known that there are an infinite number of decompositions of a 3D motion of a rigid object into rotation and translation. In our motion from stereo algorithm, the displacement of the robot or other moving objects between two successive frames is decomposed into a rotation about an axis through the origin, followed by a translation. That is, for all primitives in correspondence \((P_1^i\) in the first frame and \(P_2^i\) in the second one, \(i = 1 \ldots n)\), we have the following relation:

\[
P_2^i = R_i P_1^i + \vec{c}
\]

where \(R\) is the rotation matrix and \(\vec{c}\) is the translation vector. The following question should be answered:

Given the motion parameters \(R\) and \(\vec{c}\) in stereo coordinate system, what is the real displacement of the robot?

If we have no knowledge about the kinematics of the robot, we cannot answer the question. [10] have proposed an algorithm to recover the center of rotation when the object of interest undergoes a precession movement. This is not the case for our robot. It usually moves on a plane, that is, the rotation axis theoretically does not change direction during a short term motion. Thus the rotation center is unrecoverable by their technique.

To simplify the problem, we suppose that the robot first rotates around an axis, then translates. If the axis goes through the origin, the displacement of the robot can be characterized by \(R^t, -\vec{c}\) (in the second frame), where \(R^t\) denotes the transpose of \(R\). Suppose the axis passes through a point \(C\) (the center of rotation in a local stereo system, see Figure 4). Let \(R, \vec{c}\) be the rotation and translation of the robot and \(I\) be the 3×3 identity matrix. Then we have the following equations:

\[
R_r = R^t
\]

\[
\vec{c}_r = C_2 - C_1 = C_1 - (RC_1 + \vec{c}) = (I - R)C - \vec{c},
\]

where \(C_1\) is the coordinate vector of \(C\) in the first frame, \(C_2\) that in second frame, and \(C_1^r\) is the corresponding coordinate vector of \(C_1\) in the second frame (i.e., \(C_1^r = RC_1 + \vec{c}\)).
To meet the requirements of navigation, we must recover the rotation center C and the translation direction of the mobile robot. If the robot makes a pure rotation, that is $R = 0$, from Equation 3 we have

$$\mathbf{R} \mathbf{C} = \mathbf{C}$$

(4)

where $\mathbf{R}, \mathbf{C}$ are the displacement between the two 3D views, estimated by our "motion from stereo" algorithm which we shall summarize in the next section. But from Equation 4, we can not completely solve for $\mathbf{C}$, because $\text{Rank}(\mathbf{I} - \mathbf{R}) \leq 2$. Indeed, if the robot undergoes a pure translation, then $\mathbf{R} = \mathbf{I}$, i.e., $\text{Rank}(\mathbf{I} - \mathbf{R}) = 0$. If the motion includes rotation, there exists a vector $\mathbf{r}' \neq 0$ (the direction vector of the rotation axis) such that $\mathbf{R} \mathbf{r}' = \mathbf{r}'$, i.e., $\text{Rank}(\mathbf{I} - \mathbf{R}) = 5$. This is in agreement with our earlier remark. Indeed, Equation 4 defines the rotation axis. In practice, it is sufficient to recover a point on the rotation axis. The intersection point between the line given by Equation 4 and a plane parallel with the ground plane (we choose $y = 0$ for simplicity) can be taken as point $\mathbf{C}$. The direction of the rotation axis $\mathbf{r}'$ between the two view motion is also available.

The translation direction can be easily recovered. Let the robot undergo a pure translation, i.e., $\mathbf{R} = \mathbf{I}$. From Equation 3, we have $\mathbf{T} = -\mathbf{C}$, and the direction of translation is

$$\mathbf{t} = \frac{\mathbf{r}}{||\mathbf{r}||} = -\frac{\mathbf{C}}{||\mathbf{C}||}$$.

Up to now, we have had an estimate of real calibration parameters ($\mathbf{F}, \mathbf{C}, \mathbf{r}'$). In order to reduce the uncertainty of these parameters due to the vision system and especially due to the mechanical system, we let the robot make several pure rotations and several pure translations to compute several estimates ($\mathbf{F}, \mathbf{C}, \mathbf{r}'$), and carry out a least-squares adjustment to obtain a more precise estimate. Now we have $\mathbf{r}'$, the direction of the rotation axis, $\mathbf{C}$, a point on the rotation axis and $\mathbf{t}$, the translation direction.

4 Displacement Estimation from Stereo

In navigation, given a sequence of stereo frames reconstructed in different positions, we must register them and determine the motion between them. In this section, we present the principle of the algorithm that estimates the robot displacement from stereo frames (sets of 3D segment). The details can be found in [2,11].

The algorithm is divided into two stages. In the first stage, called generation, the rigidity constraint is heavily used to generate hypotheses of matches between two successive stereo frames. As illustrated in Figure 5, let $AB, CD$ be two segments in the first frame, $A'B', C'D'$ in the second frame, and $M_1, M_2, M'_1$ and $M'_2$ be their midpoints. If these two pairings of segments satisfy the following constraints:

$$|| AB || - || A'B' || < \epsilon_{11}$$

$$|| CD || - || C'D' || < \epsilon_{12}$$

$$| \cos(AB, CD) - \cos(A'B', C'D') | < \epsilon_{41}$$

$$|| M_1 M_2 || - || M'_1 M'_2 || < \epsilon_{42}$$

$$| \cos(AB, M_1 M_2) - \cos(A'B', M'_1 M'_2) | < \epsilon_{43}$$

then they are considered as a plausible hypothesis. Thresholds $\epsilon_{11}, \epsilon_{12}, \epsilon_{41}, \epsilon_{42}, \epsilon_{43}$ are determined dynamically using the covariance matrices on the parameters of the 3D segments.

If we take every two possible pairings in two frames, the complexity of this phase is $O(pq)$, where $p$ and $q$ are the numbers of segments in the first and second frames. A number of strategies are exploited to reduce the complexity of the generation algorithm:

Sort the segments Sort all segments in each frame in decreasing length order, so that we can easily find, by binary search, the segments in the second frame which are compatible in length with the segments in the first one.

Control search depth

Rather than find all possible matches compatible with a certain estimation of displacement, we can stop if we have found a sufficient number of compatible pairings ($\delta$, for instance).

Avoid redundant hypotheses If a pairing is already retained as a potential match in some early hypothesis, further search is not required, because it does not give us new information about the motion between two frames.
Reduce search width Consider segments of the first frame only in the central part of the frame, because segments on the sides are likely to move out of the view field in the next frame.

Reduce the number of segments Choose only the longest segments in the first scene, for instance, the p/n longest (n = 1, 2 or 3).

This algorithm does not use any information on motion such as that from the odometric system (of course, we should first calibrate before we use this information). Indeed, we exploit one general constraint on motion — that the robot moves horizontally — to speed up further our generation process.

In the second stage, called verification, we propagate each hypothesis over the whole frame. We can obtain an initial estimate of displacement from each hypothesis using the iterative Extended Kalman Filter [4]. We apply this estimate to the first frame and compare the transformed frame with the second frame. If a transformed segment of the first frame is similar enough to a segment in the second frame, this pairing is considered as matched and the Extended Kalman Filter is again used to update the displacement estimation. The similarity of two segments is discussed in (11). After all segments have been processed, we obtain, for each hypothesis, the optimal estimate, the estimated error given by the filter and the number of matches. Once all hypotheses are evaluated, the hypothesis which gives the minimal estimated error and the largest number of matches is considered as the best one, and its corresponding optimal estimate is retained as the displacement between the two frames.

In our algorithm, a rotation is defined by a 3 dimensional vector \( \hat{r} \) whose norm is equal to the rotation angle and whose corresponding normalized vector is the same as the rotation axis.

This algorithm is easily extended to determine the multiple-objects motion as detailed in [11].

5 Navigation Problem

Using the above algorithm and the results of Section 3, we can recover \( \hat{r}, C \) and \( \hat{f} \) of the mobile robot. Now, the navigation problem can be solved.

By using a stereo-vision system as a perception tool to interact with the environment, the robot can determine the next position of interest to visit [8]. More precisely, if the robot must go to another position and orientation (from position \( A \) to \( B \) in Figure 6) described by \( R \) and \( \hat{f} \) in the stereo coordinate system, this can be achieved in three steps:

- Rotate first to the required direction of translation \((\hat{C}C' = \hat{d} \text{ in Figure 6})\),
- Translate by \( \hat{d} \)
- Rotate again to the required orientation of the mobile robot.

Here we suppose that no obstacles exist between \( A \) and \( B \). The translation \( \hat{d} \) can be computed as follows:

\[
\hat{d} = RC + \hat{f} - C = (R - 1)C + \hat{f}
\]

So the first rotation angle \( \theta_1 \) is given up to its sign by

\[
\theta_1 = \cos^{-1}(\hat{f} \cdot \hat{d})
\]

where \( \cdot \) is the scalar product of vectors. The rotation angle \( \theta_1 \) is defined positive if \( \hat{f}, \hat{d} \) and \( \hat{r}' \) satisfy the right rule, i.e., \( \hat{r}' \) is parallel to \( \hat{f} \) cross \( \hat{d} \) and negative if they do not.

Theoretically, the rotation axis \( \hat{r}' \) defined by \( R \) is parallel to \( \hat{f} \). In practice, there is a slight difference. We can project \( \hat{r}' \) on \( \hat{f} \) to compute the total rotation angle \( \theta \). Then the second rotation angle \( \theta_2 = \theta - \theta_1 \).

So, with the following three commands, the robot accomplishes its displacement of \( R \) and \( \hat{f} \) from position \( A \) to position \( B \):

\[
\text{MOVE P RC= R}_1,-R_1 P=2|R_1|
\]

\[
\text{MOVE P RC= -}[\hat{A}B], -|\hat{A}B| P=|\hat{A}B|
\]

\[
\text{MOVE P RC= } R_2,-R_2 P=2|R_2|
\]

where \( R_1 = \theta_1 \times 900/\pi \) and \( R_2 = \theta_2 \times 900/\pi \).

6 Experimental Results

In this section, a complete visual navigation loop is presented. Such a loop brings together the techniques developed earlier and provides an interesting test for the vision algorithms.

The calibration is done by three pure rotations and only one translation. The rotation angles and the translation distance are of little importance, since we use the \( R \) and \( \hat{f} \) given by the motion estimation algorithm. We carry out the least-squares adjustment only on the parameters \( \hat{f} \) and \( C \). The calibration results are displayed as follows:

309
The calibration results are used in the visual navigation demonstration. The robot is first in position A. It takes three images (see Figure 7) and Figure 8 shows the 3D reconstructed frame in that position. Then we move the robot to another position B with about 10 degrees of rotation and 750 millimeters of translation, where the robot again takes three images (see Figure 9). The corresponding 3D frame is reconstructed and is displayed in Figure 10. The triangle in
Figure 9: Three images taken by the robot in position B.

Figure 10: Front and top views of the 3D frame in position B each frame is the optical center of our trinocular system. The objective of this demonstration is to recover the initial position A from position B and to return to position A. Note that there is a large displacement between these two positions, which can be noticed by simply superimposing the two frames (see Figure 11).

The motion estimation algorithm is applied to these two 3D frames and the displacement is recovered:

<table>
<thead>
<tr>
<th></th>
<th>x component</th>
<th>y component</th>
<th>z component</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Delta )</td>
<td>-5.427e-03</td>
<td>-1.625e-01</td>
<td>+1.268e-03</td>
</tr>
<tr>
<td>( \epsilon )</td>
<td>+3.057e+02</td>
<td>-2.519e+00</td>
<td>+6.887e+02</td>
</tr>
</tbody>
</table>

From \( \Delta \) we know that there is \(-9.32\) degrees of rotation and that the direction of the rotation axis is \((-5.826e-04, -1.744e-02, 1.361e-04)\), almost vertical. We apply this final
From these results, three robot commands are generated (as described in Section 5) and are executed:

**MOVE P RC=96, -96 P=192** i.e., 19.1 degrees of rotation

**MOVE P RC=-747, -747 P=747** i.e., 747 millimeters of translation

**MOVE P RC=-142, 142 P=285** i.e., -28.4 degrees of rotation

That is, the total rotation is -9.3 degrees. The robot is now in position A', not far from A. To measure the error of this demonstration, the robot takes again three images (see Figure 13). The corresponding 3D frame is reconstructed in this position (see Figure 14) and the motion estimation algorithm is applied again to the 3D frames of A and A'. We obtain:

<table>
<thead>
<tr>
<th>x component</th>
<th>y component</th>
<th>z component</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.481e-04</td>
<td>-1.966e-02</td>
<td>2.762e-06</td>
</tr>
<tr>
<td>2.487e+01</td>
<td>-2.206e-01</td>
<td>-8.83e+00</td>
</tr>
</tbody>
</table>

That is, we have an error of rotation equal to 0.6 degrees and an error of translation equal to 27 millimeters, both less than
10%. Figure 15 shows the difference between the two frames. Note that this error is the cumulation of errors in all different phases of our navigation loop, particularly the error of the mechanical system of the robot.

Acknowledgement

The authors would like to thank X. Chen and R. Vistnes for their help in correction of this paper and to thank anonymous reviewers for their comments.

References


